Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 3

Тема Построение и программная реализация алгоритма многомерной интерполяции табличных функций.

Студент Никуленко И.В.

Группа ИУ7-42Б

Оценка (баллы) _______________

Преподаватель Градов В.М.

Москва.
2021 г

Цель работы. Получение навыков владения методами интерполяции таблично заданных функций с помощью кубических сплайнов.

1 Исходные данные

1. Таблица функции с количеством узлов N. Задаётся с помощью формулы y = x^2 в диапазоне [0..10] с шагом 1.
2. Значение аргумента x в первом интервале, например, при x=0.5 и в середине таблицы, например, при x= 5.5. Сравнить с точным значением.

2 Код программы

Код программы представлен на листингах 1-3.

Листинг 1. newton.py

```
from math import fabs, ceil

def point_selection(table, n, x, flag_tab=True):
    table_size = len(table)
    if flag_tab:
        closest_point_indx = min(range(table_size), key=lambda i: abs(table[i][0] - x))
    else:
        closest_point_indx = min(range(table_size), key=lambda i: abs(table[i] - x))
    req_space = ceil(n/2)
    if closest_point_indx + req_space + 1 > table_size:
        start = table_size - n
        end = table_size
    elif closest_point_indx < req_space:
        start = 0
        end = n
    else:
        start = closest_point_indx - req_space + 1
        end = start + n

    return start, end

def newton(orig_table, n, x):
    table = []
    for i in orig_table:
        table.append(i[:])
    start, end = point_selection(table, n, x)
    table = table[start:end]
    for i in range(1, n):
        for j in range(n - 1, i - 1, -1):
            table[j][1] = (table[j][1] - table[j - 1][1]) / (table[j][0] - table[j - i][0])

    result = 0
    for i in range(n):
        temp = table[i][1]
        for j in range(i):
            temp *= (x - table[j][0])
        result += temp

    return result
```

Листинг 2. spline.py

```python
def straight_walk(ksi, eta, length, table, h):
    for i in range(3, length):
        F = -3 * ((table[i - 1][1] - table[i - 2][1]) /
                  h[i - 1] - (table[i - 2][1] - table[i - 3][1]) / h[i - 2])
        denominator = -2 * (h[i - 1] + h[i - 2]) - h[i - 2] * ksi[i - 1]
        ksi[i] = h[i - 1] / denominator
        eta[i] = (F + h[i - 2] * eta[i - 1]) / denominator


def forward_walk(ksi, eta, c):
    c[-2] = eta[-1]
    for i in range(len(c) - 2, 1, -1):
        c[i] = ksi[i + 1] * c[i + 1] + eta[i + 1]


def find_additional(d, b, a, c, h):
    for i in range(1, len(d) - 1):
        d[i] = (c[i + 1] - c[i]) / 3 / h[i]
        b[i] = (a[i + 1] - a[i]) / h[i] - h[i] / 3 * (c[i + 1] + 2 * c[i])


def find_section(points: list, x):
    if points[0][0] - points[1][0] < 0:
        i = 0
        while x > points[i][0] and i < len(points) - 1:
            i += 1
        return i
    else:
        i = 0
        while x < points[i][0] and i < len(points) - 1:
            i += 1
        return i


def find_result(x, a, b, c, d, xl):
    return a + b * (x - xl) + c * (x - xl) ** 2 + d * (x - xl) ** 3


def spline(table, x):
    a = [0] + [p[1] for p in table]
    h = [0] + [table[i][0] - table[i - 1][0] for i in range(1, len(table))]
    c = [0 for i in range(len(a))]
    b = c[:]
    d = c[:]
    ksi = c[:]
    eta = c[:]
    straight_walk(ksi, eta, len(c), table, h)
    forward_walk(ksi, eta, c)
    find_additional(d, b, a, c, h)
    i = find_section(table, x)
    result = find_result(x, a[i], b[i], c[i], d[i], table[i - 1][0])
    return result
```

Листинг 3. main.py

```python
from newton import newton
from spline import spline


def main():
    tab = [[i, i**2] for i in range(11)]
    x = 0.5

    print('\n\nТаблица функции (y = x^2):')
    print("+{:11s}|{:11s}+".format('-', '-').replace(' ', '-'))
    print('|     X     |     Y     |')
    print("|{:11s}|{:11s}|".format('-', '-').replace(' ', '-'))

    for i in tab:
        print('|{:11d}|{:11d}|'.format(i[0], i[1]))
        print("|{:11s}|{:11s}|".format('-', '-').replace(' ', '-'))
    print('\nX =', x)

    print('Результат интерполяции кубическим сплайном: {:.6f}'.format(spline(tab, x)))
    print('Значение y(x): {:.3f}'.format(x ** 2))
    print('Результат интерполяции полиномом Ньютона 3-ей степени: {:.6f}'.format(newton(tab, 3, x)))


if __name__ == "__main__":
    main()
```

3 Результаты работы

Значения y(x) при заданных x, сравнение результатов интерполяции кубическим сплайном и полиномом Ньютона 3-ей степени.

```
Таблица функции (y = x^2):
+-----------|-----------+
|     X     |     Y     |
|-----------|-----------|
|          0|          0|
|-----------|-----------|
|          1|          1|
|-----------|-----------|
|          2|          4|
|-----------|-----------|
|          3|          9|
|-----------|-----------|
|          4|         16|
|-----------|-----------|
|          5|         25|
|-----------|-----------|
|          6|         36|
|-----------|-----------|
|          7|         49|
|-----------|-----------|
|          8|         64|
|-----------|-----------|
|          9|         81|
|-----------|-----------|
|         10|        100|
|-----------|-----------|

X = 0.5
Результат интерполяции кубическим сплайном: 0.341506
Значение y(x): 0.250
Результат интерполяции полиномом Ньютона 3-ей степени: 0.250000

X = 5.5
Результат интерполяции кубическим сплайном: 30.250345
Значение y(x): 30.250
Результат интерполяции полиномом Ньютона 3-ей степени: 30.250000
```

4 Вопросы при защите лабораторной работы

1. Получить выражения для коэффициентов кубического сплайна, построенного на двух точках.

$N = 1$, так как даны только две точки.

$a = y_0$

Положим, что вторая производная на концах участка интерполирования равна нулю, тогда $c = 0$, $d = 0$

$b = (y_1 - y_0) / (x_1 - x_0)$

2. Выписать все условия для определения коэффициентов сплайна, построенного на 3-х точках.

   1. Совпадение значения сплайна и интерполируемой функции в 3=х точках.
   2. Равенство в точке 1 первой и второй производных, вычисляемых по коэффициентам на соседних участках.
   3. Равенство второй производной нулю на краях участка интерполирования.

3. Определить начальные значения прогоночных коэффициентов, если принять, что для коэффициентов сплайна справедливо $C_1=C_2$.

$c_1 = \xi * c_2 + \eta \quad \Rightarrow \quad \xi = 1, \eta = 0$

4. Написать формулу для определения последнего коэффициента сплайна C_N, чтобы можно было выполнить обратный ход метода прогонки, если в качестве граничного условия задано $kC_{N-1}+mC_N=p$, где k,m и p - заданные числа.

$C_{N-1} = \xi_N * C_N + \eta_N$,

$kc_{N-1} + mc_N = p \quad \Rightarrow \quad C_N = (p - k * \eta_N) / (k * \xi_N + m)$